モンテカルロ木探索における確定勝利優先アルゴリズム

伊藤 毅志、佐藤 慎太郎
電気通信大学 情報工学科

コンピュータ囲碁において、モンテカルロを用いた木探索アルゴリズムが成功を収めている。しかし、チェスライクゲームでは、この手法であまり大きな成果が上がっていない。そのひとつの原因として、探索の結果勝敗が決するような局面でも、その手を選ばなかったときの確率を過大評価してしまうことによる誤った勝敗確率計算の問題が挙げられる。本研究では、探索の結果、勝利が確定的になる局面をカットすることにより、正確な勝敗確率を計算するアルゴリズムを紹介する。

Deterministic victory priority algorithm in the Monte-Carlo tree search

Takeshi Ito, Shintaro Sato
Department of Computer Science, The University of Electro-Communications

In Computer Go, using Monte Carlo tree search algorithm was crowned with epoch-making success. However, in chess-like games this technique has not been used. In the position where victory or defeat is decided as a result of search, overestimating the probability when not choosing the move is the one cause. In this research, we explain the algorithm which calculates exact victory-or-defeat probability by cutting the move where a victory becomes deterministic as a result of search.

１．はじめに

近年、コンピュータ囲碁の分野ではモンテカルロ法がひとつの大きなブレークスルーとなっている。

2006 年に Computer Olympiad の９路部門で Crazy Stone が優勝し、翌 2007 年には１９路部門で MoGo が優勝をさらいモンテカルロ法に注目が集まった。また、同年電気通信大学で開催された UEC 杯コンピュータ囲碁大会においても、Crazy Stone が優勝し、この手法を用いたプログラムが大きな成果を収めている[1]。

今年の 3 月末には、パリ囲碁トーナメントのエキシビションマッチで初の公の場でのプロ対コンピュータの対戦が実現し、9 路盤で MoGo がタラヌ・カタリン五段に１勝を挙げた。また国内でも、９月に行われた FIT2008 のイベントにおいて、Crazy Stone がプロ棋士の青葉かおり四段に 19 路盤で８子ハンデ戦ながらも勝利を収め、この数年で飛躍的に棋力を伸ばしている[2]。

しかし、モンテカルロ法を将棋のようなチェスライクゲームに応用しようとする研究は少ない。これは、チェスライクゲームでは探索アルゴリズムの研究が進んでおり、モンテカルロ法を使うまでもなく、既に一定の成功を収めているという事情がある。それだけではなく、囲碁と将棋ではゲームとしての違いがあり、将棋ではモンテカルロ法のような手法が向かないと考えられているというこ

ともある。たとえば、囲碁では合法手の中で打つとマイナスになる手は少ないが、将棋は打つとマイナスになる手が圧倒的に多い。乱数で手を選ぶモンテカルロ法がプレイアウトの時に正解手順をたどる可能性がきわめて低く、このことがモンテカルロ法は適さないと考えられている一要因になっている。

特に終盤の局面などでは、たくさんの変化手順の中で、唯一の勝ち手順以外は負けとなってしまうような局面が将棋では非常によく現れる。このような局面において勝敗確率のみで局面を評価すると、唯一の勝ち手順以外の多くの負け手順確率が邪魔をして、正確な手の判断ができなくなってしまうといった問題点がある。

そこで本研究では、勝ち手順が見つかった場合には、UCT 探索の過程でその結果を後ろ向きに伝播させ、勝ち手順のみを優先させるアルゴリズムを提案し、この問題の解決を試みた。

２．５五将棋への MC-UCT 法の適用

基本的な思考アルゴリズムは、コンピュータ囲碁で用いられている UCT を使ったモンテカルロアプローチとなっている。

本研究では、将棋と同様の性質を持つ５五将棋を題材に研究を行った。

チェスライクゲームへの適用において問題になるのは、終局手数が一定でないということが挙げられる。囲碁は終局までの手数が概ね想定されるが、将棋ではランダム対局では決着がつかない可能性がある。そこで、我々は UCT を組み込んだランダムシミュレーション実験を１６手、２４手、３２手、４０手、４８手と変化させながら、思考アルゴリズム同士の自己対戦を２０局ずつ行って勝率を比較した。プレイアウトまでに勝った時には 1、負けた時には 0、引き分けの時には 0.5 を返すようにしてシミュレーションした。表１はその結果である。表中の数字は、（勝数－負数）を表している。

この結果から、４０手で終了させたものがもっとも好成績を収めた。したがって、以降は４０手で終了させるアルゴリズムを採用した。一般に人間同士の対局でも 40 手程度で決着することが多く、この結果はその事実とも合致する。

	後手から見た勝ち数						
	終了手数	16	24	32	40	48	先手勝敗
先手から見た勝ち数	16		−3	−3	−2	−1	−9
	24	−1		1	−4	−1	−5
	32	3	−1		−3	0	−1
	40	5	−1	3		0	7
	48	3	3	−1	1		6
	後手勝敗	−10	2	0	8	2	
	終了手数	16	24	32	40	48	
	合計勝敗	**−19**	**−3**	**−1**	**15**	**8**	

表１　終了手数決定のための自己対戦

また、囲碁で手の選択でパターン知識を用いていたように、このアルゴリズムでも、事前知識である手の分類を２カ所で利用している。１つは、モンテカルロシミュレーションにおける手の生起確率として用いており、もう１つは、ゲーム木の選択処理のときに、どの手から優先的に調べるかを決める着手緊急度(FPU)の値を決めるために用いている。

カテゴリ	手の分類	倍率
A1	歩を取る手	10 倍
A2	歩以外を取る手	20 倍
B1	利きのないところに移動する手	10 倍
B2	利きの勝っているところに移動する手	10 倍
B3	利きのないところに打つ手	20 倍
B4	利きの勝っているところに打つ手	10 倍
C	相手の陣地に飛車を打つ手	10 倍
D	相手の王の筋に角を打つ手	10 倍
E	王手をかける手	30 倍

（同じカテゴリの項目は重複して適用しない）

表２　手の分類と選択確率

「手の分類」の一覧とその影響度を表 2 および表 3 に示す。分類とその倍率・FPU の値はシミュレーションを繰り返して最適な値を経験的に求めた。

倍率	FPU 値
6000 倍	0.90 + U
3000 倍	0.85 + U
600 倍	0.70 + U
300 倍	0.60 + U
200 倍	0.55 + U
100 倍	0.45 + U
30 倍	0.35 + U
20 倍	0.30 + U
10 倍	0.25 + U
1 倍	0.20 + U

$$U=\sqrt{\frac{2\log 16}{16}}$$

表３　手の倍率とＦＰＵ値

この修正された値を用いることで、UCT アルゴリズムは非常に効果的に働き、UCT を用いないアルゴリズムに対して９９％の勝率を挙げることができた。

３．確定勝利優先アルゴリズム

３．１　モンテカルロ法の問題点

知識と UCT を組み込むことで、棋力は飛躍的に向上したが、特定の場面でモンテカルロ法の弱点が見られた。

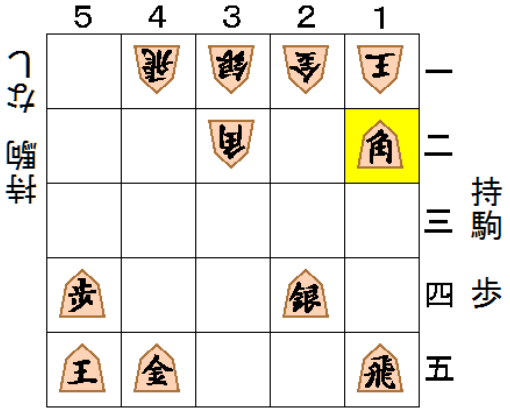

図１　従来のＭＣ法で誤りやすい局面例

図１の局面は、UCT を用いたモンテカルロ法プログラムでは、なかなか正解にたどり着けない例である。この局面は、初期局面から３八角、３二角、２四銀、４一飛、１二角と５手進んだ局面である。

この局面で、後手が同金と取ると、以下、同飛、同玉、１三歩、２一玉、１二金と詰んでしまう。しかし、実際にはこの局面で UCT アルゴリズムを組み込んだプログラムは、同金と取ってしまい、正確に自玉の詰みを読み切れなかった。

このような局面で、なぜ正しい手が選ばれなかったのかを調べたところ、詰ます方の合法手が非常に多く、しかも詰ます手以外の手を選んだ場合大きな駒損となり、負けに直結する局面であったことがわかってきた。このような場合、UCT アルゴリズムでは、正解手で詰みを発見したとしても、それを単なる１勝としか判断せず、他の不正解の手も同様に調べてしまう。不正解の手を選んだ場合の勝率が低いため、折角の正解手順の１勝が打ち消されてしまう。十分な回数のプレイアウトが行われれば、いずれ正解手順のみが生き残るはずだが、そのためには相応の回数のプレイアウトが必要である。

３．２　確定勝利優先アルゴリズム

上述のような問題点を解消するために、本研究では以下のようなアルゴリズムを導入することにした。

将棋は囲碁と違い、途中で確実にゲームが終了する状況（すなわち詰み）が存在する。そのような状況のノードを見つけたら、そのノードに「勝ちフラグ」または「負けフラグ」を記す。「勝ちフラグ」がたつ条件とは、そのノードから行ったモンテカルロシミュレーションの一手目で相手を詰めた時である。「負けフラグ」がたつ条件とは、そのノードが既に負けている状態（合法手がない状態）だったときである。そして、ゲーム木の選択処理を行う際に、「負けフラグ」がついているノードへ直接つながっているノードを見つけたら、そのノードに「勝ちフラグ」を立て、以降の選択を行わず、勝利であることを親ノードに伝える。逆に、「勝ちフラグ」がたっているノードは、その評価値に関わらず選択しないようにする。もしすべての子ノー

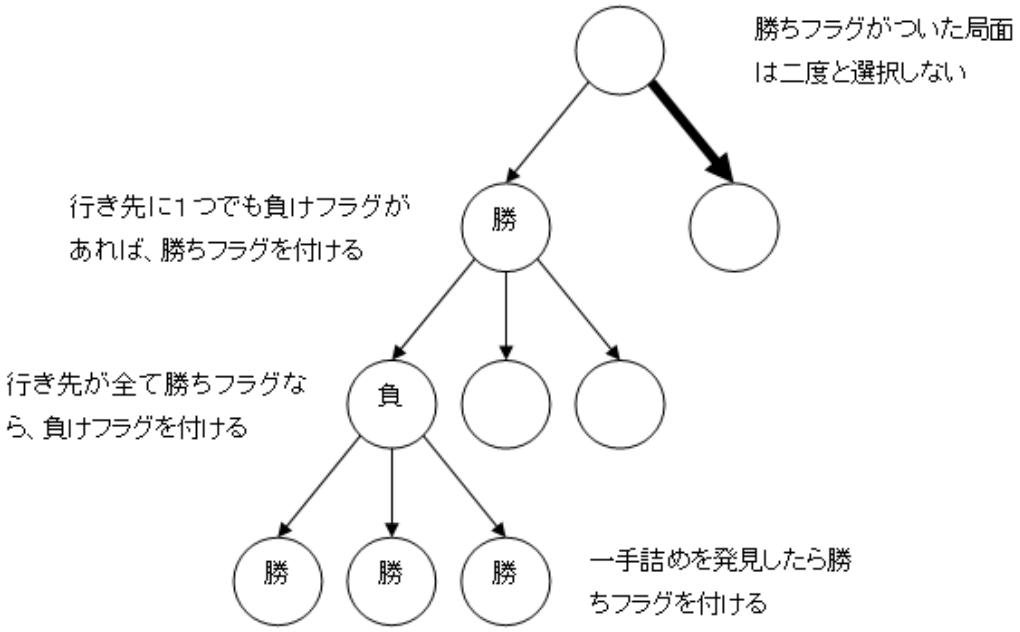

図２　確定勝利優先アルゴリズムの処理イメージ

ドに「勝ちフラグ」が立っている場合、そのノードに「負けフラグ」を立て、敗北であることを親ノードに伝える。この処理のイメージを図２に示す。この処理を実装することで、確実に負けることに気づいた時点ではそのノードを探索しなくなり、詰みに関する探索効率が向上することが期待される。

実際にこの処理を行うように改良したUCTアルゴリズムで、図１の局面を探索させたところ、３００回程度のプレイアウト回数で同金では詰むことがわかり、その手を回避することができるようになった。

４．おわりに

本報告では、チェスライクゲームの一つである５五将棋にモンテカルロ法を適用する手法について説明した。そして、問題点の一つとして、正確な詰み手順を読み切ることの難しさを指摘した。そして、その問題を解消するために、確定勝利優先アルゴリズムを提案してその可能性を示した。

この手法は、詰め将棋にとどまらず、囲碁のシチョウなどの長手数の読みにも適用されるのではないかと考えている。たとえば、長手数後に確実に勝勢になる手が見つかった場合には、「勝勢フラグ」を立てることで同様の処理を行える可能性がある。これらの点についても、実証していきたい。

参考文献

[1] 伊藤毅志：第１回ＵＥＣ杯コンピュータ囲碁大会報告、情報処理学会誌、Vol.49,No.6 (2008).
[2] FIT2008「コンピュータ囲碁最前線報告」、http://homepage1.nifty.com/ta_ito/fit2008/fit2008-igo.htm